Bitwise Operator

Bitmanipulation

^	XOR	✓
&	And	✓
|	OR	✓
<<	Leftshift	
>>	Right Shift	
~	Not	

XOR → exclusively OR

↓

one bit

$1 \wedge 0 = 1$

$0 \wedge 1 = 1$

$0 \wedge 0 = 0$

$1 \wedge 1 = 0$

$5 \,\&\, 7 = 5$

$$\begin{array}{rccc} & \boxed{1} & 0 & 1 \\ \& & \boxed{1} & 1 & 1 \\ \hline & 1 & 0 & 1 \end{array}$$

$$5 \,|\, 6 = \begin{array}{ccc} 1 & 0 & 1 \\ 1 & 1 & 0 \\ \hline 1 & 1 & 1 \end{array} = 7$$

$$5 \wedge 7 = \begin{array}{rccc} & 1 & 0 & 1 \\ \wedge & 1 & 1 & 1 \\ \hline & 0 & 1 & 0 \end{array} = \boxed{2}$$

$$\begin{array}{rc} & 1 \\ + & 1 \\ \hline & 10 \end{array} \qquad \begin{array}{rc} & 1 \\ \text{OR} & 1 \\ \hline & 1 \end{array}$$

Addition OR

Left Shift

$5 << \textcircled{2}$

$= 5 * 2^2 = 5 \times 4$

$= \boxed{20}$

$\rightarrow$ $\overleftarrow{1\ 0\ 1}$

$\underset{16}{1}\ \underset{8}{0}\ \underset{4}{1}\ \underset{2}{\underline{0}}\ \underset{1}{\underline{0}} = \textcircled{20}$

Memory

32 bit

| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

$\longleftarrow$

$\longleftarrow$

0 0 0 1 1 0 0 0 0

$a << 3$

$= a * 2^3$

$a << 1$

$= \boxed{2} * a$

Right Shift (=>)

$a = 20$

$b = 2$

$$\begin{array}{ccccccc} & & 16 & 8 & 4 & 2 & 1 \\ 0 & 0 & 1 & 0 & 1 & \cancel{0} & \cancel{0} \end{array}$$

$\Rightarrow 101$

$= 5$

$a >> b$

$$20 >> 2 = \frac{20}{2^2}$$

$$= \frac{20}{4} = \boxed{5}$$

$$\boxed{a >> b = \frac{a}{2^b}}$$

Java

use this

>>

Signed Right Shift

Filler elemend = MSB

0 +ve

1 -ve

|0|0|0|0|0|0|1|0|X|

↓ >>1

filler element

|0|0 0 0 0 0 0 1 0|

|1 1 1 1 1 1 1 1 1 0 1 1|

↓ >>1

|1| 1 1 1 1 1 1 1 1 1 1 0 1|

filler element

-5 >> 2

= -2

exclusive to Java

>>>

Unsigned Right Shift

Fillcr element is always 0

5 >>> 1 = 2

-5 >>> 1 = ?

+ve

|1 1 1 1 1 1 1 1 1 0 1 1|

↓ >> 1

|0| 1 1 1 1 1 1 1 1 1 1 0 1|

filler element is always 0 irr of sign bit

huge +ve No

Java

+ve No "Signed format" → first bit always denotes sign.

|0|0|0|0|0|0|1|0|1|

32 bit

+5 Yes

-ve

|1|0|0|0|0|0|1|0|1|

32 bit

Not -5

(2s compliment form)

Convert -5 → +5 into 2s compliment form

|0|0|0|0|0|0|1|0|1|

1 1 1 1 1 0 1 0

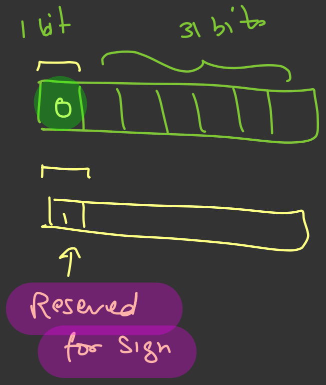
+ 1

|1 1 1 1 1 0 1 1| -5

32bit

Java (only signed int)

Signed datatype {
int x = +5,
int y = -5,



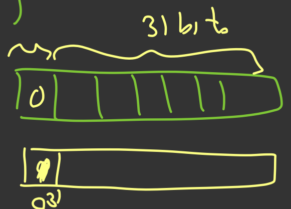
C++

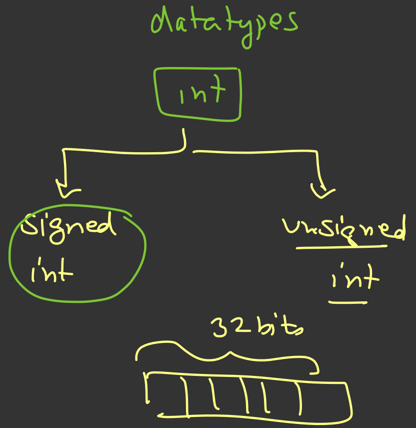
→ int x ; (signed int)

→ unsigned int x,

↓

only +ve





Python

No limit

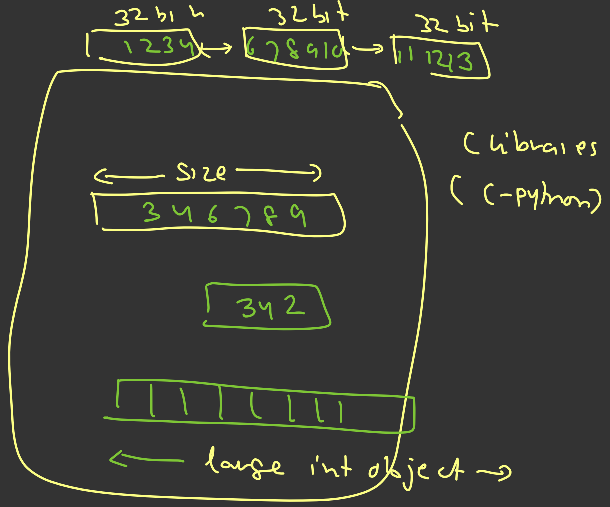
$x = 1000$

$X = 10^{1000}$ (1000 digit)

$x = +\infty$

Right → $\underline{x >> 2} \Rightarrow \frac{x}{2^2}$

Left → $\underline{x << 2} \Rightarrow x * 2^2$



- Python handles int objects very differently from Java/C++

- No overflow

Q1 $-5 >> 2$

Q2 $-5 >>> 2$

5 0 0 0 0 0 1 0 1

-5 ⇒ 1 1 1 1 1 0 1 1 →

$-5 >> 2 = -2$

Signed shift ⇒ $\underbrace{\boxed{1}\,1}_{}$ 1 1 1 1 1 0 → -ve No -2

filler is 1 as the no is negative

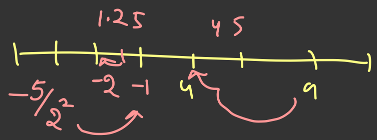
flip all bit

⇒ 0 0 0 0 0 0 0 1

$+ \quad 1$

0 0 0 0 0 0 1 0 = 2

↑



-2

$5 >> 2 = 1$

floor

$-5 >> 2 = (5/2^2)$

$= -2$

$-10 >> 3 = -2$

$\frac{-10}{2^3} = \left\lfloor \frac{-10}{8} \right\rfloor = \lfloor -1.xx \rfloor$

$= -2$

public class Main{ publ

$10 >> 3 = \left\lfloor \frac{10}{8} \right\rfloor = \lfloor 1.xx \rfloor$
$= 1$

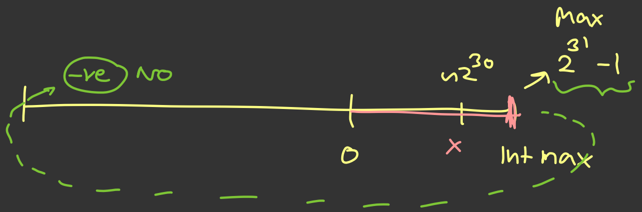
$\text{floor}(1.26) = 1$ $\qquad \boxed{1} < 2$

$\text{floor}(-1.26) = -2$ $\qquad \boxed{-2} < -1$

↑
Floor

$1\,\boxed{0}\,0\,1\,1\,0\,0\,1$

$1\,1\,0\,0\,1\,1\,0$



$x = 2^{30}$

$\boxed{11} = 2^2 - 1 = 3$

$111 = 2^3 - 1 = 7$

$1111 = 2^4 - 1 = 15$

$\underbrace{\_\ \_\ \_\ \_\ \_\ \_\ \_}_{31} = 2^{31} - 1 = \_$

$x << 3 = x \times 2^3$
$= 2^{30} \cdot 2^3$
$= \underline{\underline{2^{33}}}$

$-5 >>> 2$ // Unsigned Right Shift

$-5 \Rightarrow \underline{\textcircled{1}\ 1\ 1\ 1}\ 1\ 0\ 1\ 1 \longrightarrow$

$x = 5$

$x = x + 100$

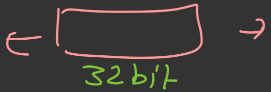
32 bit $\textcircled{-5}$

_ _ _ _ _ _ _ _ _ _ _ _ _

1 1 1 1 1 1 1 1 1 0 $\cancel{1}$ $\cancel{1}$

$\underline{0}\ \underline{0}$ 1 1 1 1 1 1 1 1 1 1 0

2^{31} $\underbrace{2^{30}\ 2^{29}\ \dots\dots\ 2^{3}\ 2^{2}\ 2^{1}}$ 2^{0}



$2^1 + \dots\dots 2^{30}$

$= 10732\dots822$

$\longleftarrow$

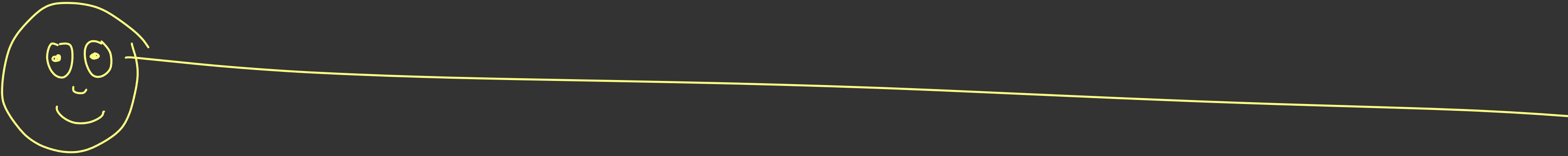
0 1 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 $\underline{0}$ $\underline{0}$ $\underline{0}$

Sout $(-5 >>> 2) \rightarrow \cancel{\dfrac{a}{2b}}$

↑

Rare



PROBLEMS

Q) Given 2N + 2 numbers, where every no repeats twice except 2 unique No's find out unique No's

[9, 3, 3, 6, 4, 4, 2, 8, 8, 9]

Algo-1 — 6, 2 — Two loops $O(N^2)$

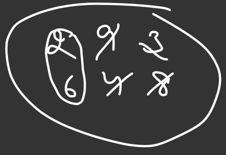
Algo-2 — Sorting — 2, 3, 3, 4, 4, 6, 8 8, 9, 9 — $O(N\log N + N)$

$= O(N\log N)$

Algo-3 — Hashmap, Hashset

9-2
3-2
2-1
4-2
6-1
8-2
9-2

$O(N)$



Hashset

$O(N)$ time
$O(N)$ Space

Algo-4 Bit manipulation

Step-1

XOR all

[~~9~~, ~~3~~, ~~3~~, 6, ~~4~~, ~~4~~, 2, ~~8~~, ~~8~~, ~~9~~]

$= 6 \wedge 2$

$= \boxed{4}$

p=2

```
  1 1 0
^ 0 1 0
-------
  1 0 0   ← alleast 1 ~~sef~~ set bit
  2 1 0
  ↑
  p=2
```

one of the no's has a set bit here

Step-2

$\overset{1001}{9}$, $\overset{011}{3}$, 3, $\overset{110}{6}$, $\overset{100}{4}$, 4, $\overset{0010}{2}$, $\overset{1000}{8}$, 8, 9

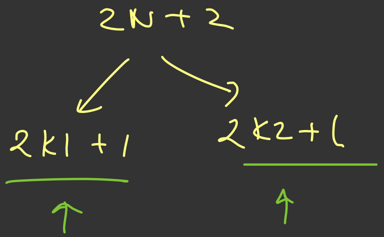
[9, 3, 3, 6, 4, 4, 2, 8, 8, 9]

↑ ↑ ↑ ↑ ↑

Step-3

$A = 6, \cancel{4}, \cancel{4}, = \boxed{6}$

$B = \cancel{9}, \cancel{3}, \cancel{3}, 2, \cancel{8}, \cancel{8}, \cancel{9} = \boxed{2}$



Code -

①

```
result = 0

for(x : arr) {

    result = result ^ x

}
```

[9, 3, 3, 6, 4, 4, 2, 8, 8, 9]

x

result = 2 ^ 6

= 4

②

Find position of 'last' set bit

```
p=0
while( (result & 1) != 0 ){
        p++
        result = result>>1;

}
```

0 0 1 0 0

p=0

0 0 1 0   p = 1

0 0 1

p = 2

$\boxed{p = 2}$

2 unique elements can be differentiated using this bit.

③ Filtering

[9, 3, 3, 6, 4, 4, 2, 8, 8, 9]

```
int A = 0
int B = 0

for( x :  arr) {
    if ((x & (1<<p)) ==0)
        A = A^x
    else
        B = B^x
}
```

Check if p^{th} bit of x is 0 or 1

$9 \rightarrow 1001$ 1<<p

$\& \; 0100$

0000 $==0 \rightarrow$ NO

$0100 > 0 \rightarrow$ Yes

$9 \rightarrow 1001$

$\& \; 0100$

0000

$3 \rightarrow 011$

$\& \; 100$

000

$4 \rightarrow 100$

010

100

000

$6 \rightarrow 110$

$\& \; 100$

100

$$A = 0 \wedge \cancel{9} \wedge \cancel{3} \wedge 2 \wedge \cancel{8} \wedge \cancel{8} \wedge \cancel{9} \wedge \cancel{3}$$

$$B = 0 \wedge 6 \wedge \cancel{4} \wedge \cancel{4}$$

$$= \boxed{2, 6}$$

O(1) Space   O(N) time

Q 3N + 1

Every no repeat twice except one unique no.

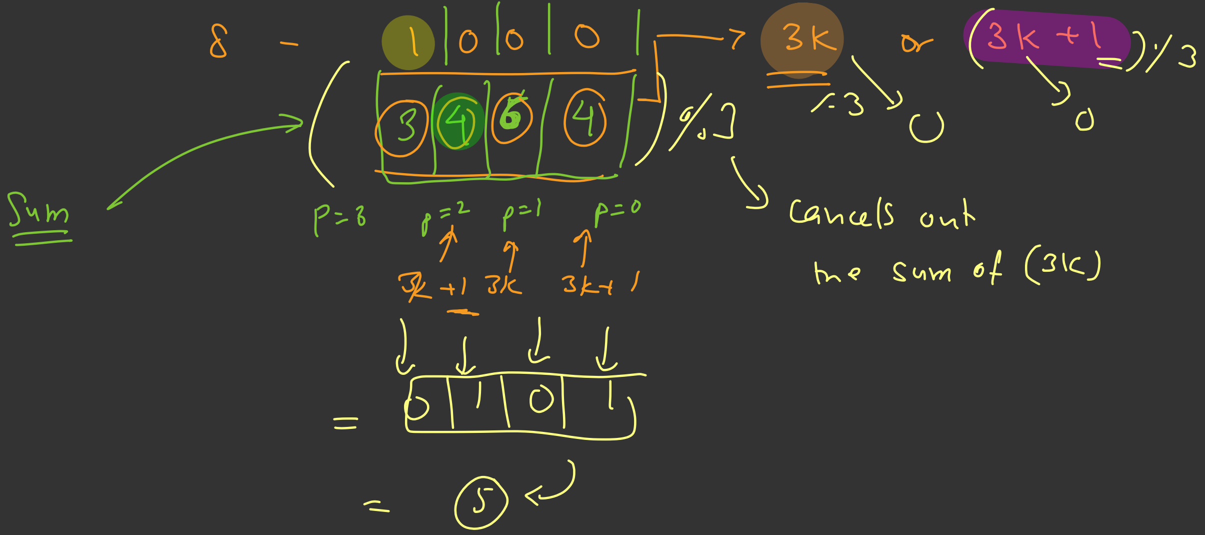
"Break"

10.32

{6, 8, 8, 5, 6, 6, 3, 3, 3, 8}

Solution → Develop a new algo s.t. no's which are repeating they get cancelled somehow.

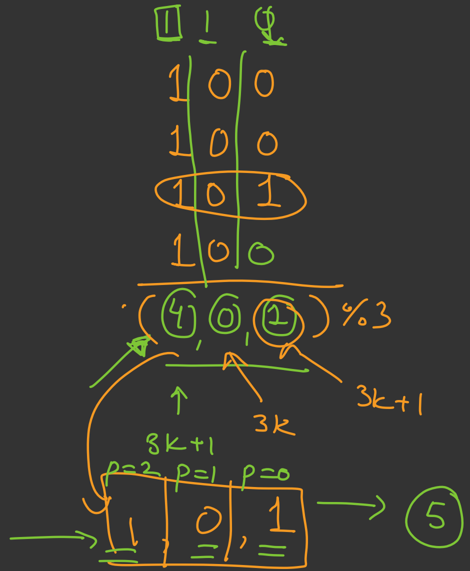
		3	2	1	0
6	–	0	1	1	0
8	–	1	0	0	0
8	–	1	0	0	0
5	–	0	1	0	1
6	–	0	1	1	0
6	–	0	1	1	0
3	–	0	0	1	1
3	–	0	0	1	1
3	–	0	0	1	1

8 -
1 | 0 | 0 | 0 | 1
3k
or
(3k + 1) % 3
% 3
0
0
3 | 4 | 6 | 4
% 3
Sum
P=3
p=2
p=1
p=0
3k+1
3k
3k+1
= 0 | 1 | 0 | 1
= 5
Cancels out
the sum of (3k)

4 , 4 , 5 , 4 , 6 , 6 , 6

↓

3



5 ⇒ 1 0 1

4 → 3

6 → 3

5 → 1

7

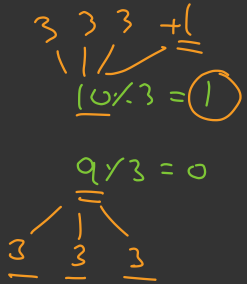
3k + 1

⇒

7 % 3 = 1

3k % 3 = 0

(3k + 1) % 3 = 1



Code

arr = [4 , 4 , 5 , 4 , 6 , 6 , 6]

Sum[32] =

0	1	2	3							31
1	3	7	0	0	0	0	0	0	0	0

```
for( int x : arr) {
    p=0
    while(x>0){
        last_bit = x&1
        sum[p] += (last_bit),
        p=p+1
        x = x>>1
    }
}
ans=0 , p=1
for(i=0, i<32, i++){
    sum[i] = Sum[i]%3
    ans = ans + sum[i]*P;
    p= p<<1
}
```

32 bits

O(1)

→ O(N) time

→ O(1) space

- 1 0 0
- 1 0 0
- 1 0 0
- 1 1 0
- 1 1 0
- 1 0 1
- 1 1 0

7

4 → 1 0 0

x

p

5 → 1 0 1

6 → 1 1 0

2^0	2^1	2^2	-----	2^{31}
1	0	1	0 0 0 0	0

1 = 2^0

ans = 0 + 1*1 + 0x2 + 1*4

print (ans);

$= 1+4$

$= \boxed{5}$

$\boxed{5N + 1}$

5, 6, 5, 5, 5, 5, 7, 7, 7, 7, 7, 7

$\boxed{\% 5}$